Math Article

A small contribution to mathematics: The matrix division

A slightly different view on solving systems of equations

Sergej Oks

Math Article

A small contribution to mathematics: The matrix division

A slightly different view on solving systems of equations

Sergej Oks

25th April 2023

Contents

1 Intention **1**

2 Derivation of the matrix division **3**
2.1 Preliminary example 4
2.2 General matrix division 7
2.3 Treatment of rectangular matrices 11

3 Conclusion **13**

4 Appendix **15**
4.1 Matlab-Code: Matrix division (original) 15
4.2 Matlab-Code: Matrix division (optimized) 16

References **17**

1 Intention

„You wanted algebra,
and you shall have your fill of it.“

Jules Verne

A so-called matrix division shall in general not exist. However, everyone knows that a system of equations can still be solved, for example by applying the Gaussian elimination method. One likes to rewrite such a problem so that the solution can be found through inversion.

$$\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b} \qquad \text{with} \qquad \boldsymbol{A} \in \mathbb{R}^{i \times j}, \boldsymbol{x} \in \mathbb{R}^{j} \text{ and } \boldsymbol{b} \in \mathbb{R}^{i} \tag{1.1}$$

$$\Leftrightarrow \qquad \boldsymbol{x} = \boldsymbol{A}^{-1} \cdot \boldsymbol{b} \tag{1.2}$$

It depends on the dimensions i and j (and the rank of $\boldsymbol{A}$) how exactly the solution looks in the end, but this only by the way. This work shall be focused on square matrices (with full rank) and, by changing the perspective, provide an approach to define the matrix division and thus to prove its existence.

2 Derivation of the matrix division

According to Einstein's summation convention the summation in the multiplication of matrices (or tensors) occurs over indices of the same name. This means that the sum sign can be dispensed with, which is very conducive to the overview of equations. This convention is used in the following and based on it an inversion is executed. Essentially, the method of matrix multiplication can be split up into three components:

- Multiplication
- Addition (Sum)
- Followed on each other (successive)

The equation 1.1 can thus be presented as follows:

$$\sum_j A_{ij} \cdot x_j = b_i \qquad \Rightarrow \qquad A_{ij} \cdot x_j = b_i \qquad \text{whereby} \qquad i, j \in \mathbb{N} \tag{2.1}$$

If you now want to solve the equation for x_j, it can be written down quite easily using Einstein's summation convention.

$$x_j = b_i / A_{ij} \tag{2.2}$$

The representation of an inversion, as in Equation 1.2, is thus unnecessary. The only question that remains is, what the interpretation and the mathematical rules to be applied here are. The thesis of this work is that the method of matrix division, i.e. the reversal of matrix multiplication, must be structured as follows:

- Division
- Subtraction (Difference)
- Nested in each other (recursive)

Equation 2.2 could thus be represented as follows,

$$x_j = b_i / A_{ij} \qquad \Rightarrow \qquad x_j = \bigtriangleup_i b_i / A_{ij} \tag{2.3}$$

where the Δ-Sign shall stand for difference. Since a difference can only ever be build between two members of an equation, this automatically results in a nested procedure in itself. The following example should clarify the process.

2.1 Preliminary example

In a 3×3 example, it shall be shown how the matrix division results from the reversal of matrix multiplication. In matrix multiplication, the elements A_{ij} are first multiplied by x_j, followed by addition respectively summation, and this for each equation in succession. In the following, the summation sign is shown explicitly in order to make it clear that it is always implicitly contained in the matrix multiplication.

$$\sum_{Columns} \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad \Leftrightarrow \quad \begin{array}{l} A_{11} \cdot x_1 + A_{12} \cdot x_2 + A_{13} \cdot x_3 = b_1 \\ A_{21} \cdot x_1 + A_{22} \cdot x_2 + A_{23} \cdot x_3 = b_2 \\ A_{31} \cdot x_1 + A_{32} \cdot x_2 + A_{33} \cdot x_3 = b_3 \end{array} \tag{2.4}$$

This procedure seems simple, but it is important to remember that it is based on our mathematical rules, including order of operations. To make it even clearer, the row $i = 1$ will be dealt with first

$$\sum_{Columns} \begin{bmatrix} A_{11} & A_{12} & A_{13} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = b_1 \quad \Leftrightarrow \quad A_{11} \cdot x_1 + A_{12} \cdot x_2 + A_{13} \cdot x_3 = b_1 \tag{2.5}$$

by multiplying the members A_{1j} by x_j and only thereafter summing them up. The index of the same name, which is used for summing, is j here. Subsequently, the same happens with the rows $i = 2$ and $i = 3$.

While multiplying you are adding up over indices of the same name, it shall now be subtracted or "differentiated" over indices of the same name when dividing. Since the index of the same name changes from j to i during the reversal (see equation 2.2), the system of equations is now developed columnwise. So the 3×3 example must look as follows.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \bigwedge_{Rows} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} / \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} = \bigwedge_{i} \frac{b_i}{\begin{bmatrix} A_{i1} & A_{i2} & A_{i3} \end{bmatrix}} \tag{2.6}$$

In order to determine x_1, the system of equations must be developed in such a way that the column $j = 1$ remains until the end. One follows the order of operations rule and divides e.g. the columns (or vectors) b_i, A_{i1} and A_{i2} first by A_{i3}.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{\bigwedge_{Rows} \begin{bmatrix} b_1/A_{13} \\ b_2/A_{23} \\ b_3/A_{33} \end{bmatrix}}{\bigwedge_{Rows} \begin{bmatrix} A_{11}/A_{13} & A_{12}/A_{13} \\ A_{21}/A_{23} & A_{22}/A_{23} \\ A_{31}/A_{33} & A_{32}/A_{33} \end{bmatrix}} = \frac{\bigwedge_{i} b_i/A_{i3}}{\bigwedge_{i} \begin{bmatrix} A_{i1}/A_{i3} & A_{i2}/A_{i3} \end{bmatrix}} \tag{2.7}$$

Alternatively, one could have divided b_i, A_{i1} and A_{i3} by A_{i2}, but you mustn't divide by A_{i1} until the end. Next, the differences between rows $i = 1$ and $i = 2$ and between rows

$i = 2$ and $i = 3$ are formed, resulting in a reduced system of equations.

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \bigtriangleup_{Rows} \frac{\begin{bmatrix} b_2/A_{23} - b_1/A_{13} \\ b_3/A_{33} - b_2/A_{23} \end{bmatrix}}{\begin{bmatrix} A_{21}/A_{23} - A_{11}/A_{13} & A_{22}/A_{23} - A_{12}/A_{13} \\ A_{31}/A_{33} - A_{21}/A_{23} & A_{32}/A_{33} - A_{22}/A_{23} \end{bmatrix}} = \bigtriangleup_{i} \frac{b'_i}{\begin{bmatrix} A'_{i1} & A'_{i2} \end{bmatrix}} \tag{2.8}$$

This process is repeated until only a scalar remains at the end. Here you can see the in itself nested calculation method, through which the system of equations is repeatedly converted into a more reduced form. For this example, this means one divides the newly resulted b'_i and A'_{i1} by the new A'_{i2}. Column $j = 1$ stays in the numerator again.

$$x_1 = \frac{\bigtriangleup_{Rows} \begin{bmatrix} (b_2/A_{23} - b_1/A_{13})/(A_{22}/A_{23} - A_{12}/A_{13}) \\ (b_3/A_{33} - b_2/A_{23})/(A_{32}/A_{33} - A_{22}/A_{23}) \end{bmatrix}}{\bigtriangleup_{Rows} \begin{bmatrix} (A_{21}/A_{23} - A_{11}/A_{13})/(A_{22}/A_{23} - A_{12}/A_{13}) \\ (A_{31}/A_{33} - A_{21}/A_{23})/(A_{32}/A_{33} - A_{22}/A_{23}) \end{bmatrix}} = \frac{\bigtriangleup_{i} b'_i/A'_{i2}}{\bigtriangleup_{i} A'_{i1}/A'_{i2}} \tag{2.9}$$

Then you form the difference between the remaining rows $i = 1$ and $i = 2$ again and get the result.

$$x_1 = \frac{(b_3/A_{33} - b_2/A_{23})/(A_{32}/A_{33} - A_{22}/A_{23}) - (b_2/A_{23} - b_1/A_{13})/(A_{22}/A_{23} - A_{12}/A_{13})}{(A_{31}/A_{33} - A_{21}/A_{23})/(A_{32}/A_{33} - A_{22}/A_{23}) - (A_{21}/A_{23} - A_{11}/A_{13})/(A_{22}/A_{23} - A_{12}/A_{13})} \tag{2.10}$$

As already mentioned above, the order of the columns by which one divides is in principle arbitrary, as long as you leave out the column j to the searched x_j until the end. This can be seen on the vector $\boldsymbol{x}$, where the remaining x_j indicate the paths that are still possible. In concrete terms, this means that the equation 2.6 can also be solved as follows.

$$\begin{bmatrix} x_1 \\ x_3 \end{bmatrix} = \frac{\bigtriangleup_{Rows} \begin{bmatrix} b_1/A_{12} \\ b_2/A_{22} \\ b_3/A_{32} \end{bmatrix}}{\bigtriangleup_{Rows} \begin{bmatrix} A_{11}/A_{12} & A_{13}/A_{12} \\ A_{21}/A_{22} & A_{23}/A_{22} \\ A_{31}/A_{32} & A_{33}/A_{32} \end{bmatrix}} = \frac{\bigtriangleup_{i} b_i/A_{i2}}{\bigtriangleup_{i} \begin{bmatrix} A_{i1}/A_{i2} & A_{i3}/A_{i2} \end{bmatrix}} \tag{2.11}$$

So one first divides by the column $j = 2$ and then forms the differences across the rows.

$$\begin{bmatrix} x_1 \\ x_3 \end{bmatrix} = \bigtriangleup_{Rows} \frac{\begin{bmatrix} b_2/A_{22} - b_1/A_{12} \\ b_3/A_{32} - b_2/A_{22} \end{bmatrix}}{\begin{bmatrix} A_{21}/A_{22} - A_{11}/A_{12} & A_{23}/A_{22} - A_{13}/A_{12} \\ A_{31}/A_{32} - A_{21}/A_{22} & A_{33}/A_{32} - A_{23}/A_{22} \end{bmatrix}} = \bigtriangleup_{i} \frac{b'_i}{\begin{bmatrix} A'_{i1} & A'_{i2} \end{bmatrix}} \tag{2.12}$$

Then you repeat the division by the newly resulted column $j = 2$,

$$x_1 = \frac{\bigwedge\limits_{Rows} \begin{bmatrix} (b_2/A_{22} - b_1/A_{12})/(A_{23}/A_{22} - A_{13}/A_{12}) \\ (b_3/A_{32} - b_2/A_{22})/(A_{33}/A_{32} - A_{23}/A_{22}) \end{bmatrix}}{\bigwedge\limits_{Rows} \begin{bmatrix} (A_{21}/A_{22} - A_{11}/A_{12})/(A_{23}/A_{22} - A_{13}/A_{12}) \\ (A_{31}/A_{32} - A_{21}/A_{22})/(A_{33}/A_{32} - A_{23}/A_{22}) \end{bmatrix}} = \frac{\bigwedge\limits_{i} b'_i/A'_{i2}}{\bigwedge\limits_{i} A'_{i1}/A'_{i2}} \tag{2.13}$$

form the differences again and get an alternative expression for x_1,

$$x_1 = \frac{(b_3/A_{32} - b_2/A_{22})/(A_{33}/A_{32} - A_{23}/A_{22}) - (b_2/A_{22} - b_1/A_{12})/(A_{23}/A_{22} - A_{13}/A_{12})}{(A_{31}/A_{32} - A_{21}/A_{22})/(A_{33}/A_{32} - A_{23}/A_{22}) - (A_{21}/A_{22} - A_{11}/A_{12})/(A_{23}/A_{22} - A_{13}/A_{12})} \tag{2.14}$$

which reflects the same result as the equation 2.10. To get the other x_j one can, for example, divide the b'_i and A'_{i2} from the equation 2.8 by A'_{i1}

$$x_2 = \frac{\bigwedge\limits_{Rows} \begin{bmatrix} (b_2/A_{23} - b_1/A_{13})/(A_{21}/A_{23} - A_{11}/A_{13}) \\ (b_3/A_{33} - b_2/A_{23})/(A_{31}/A_{33} - A_{21}/A_{23}) \end{bmatrix}}{\bigwedge\limits_{Rows} \begin{bmatrix} (A_{22}/A_{23} - A_{12}/A_{13})/(A_{21}/A_{23} - A_{11}/A_{13}) \\ (A_{32}/A_{33} - A_{22}/A_{23})/(A_{31}/A_{33} - A_{21}/A_{23}) \end{bmatrix}} = \frac{\bigwedge\limits_{i} b'_i/A'_{i1}}{\bigwedge\limits_{i} A'_{i2}/A'_{i1}} \tag{2.15}$$

to determine x_2.

$$x_2 = \frac{(b_3/A_{33} - b_2/A_{23})/(A_{31}/A_{33} - A_{21}/A_{23}) - (b_2/A_{23} - b_1/A_{13})/(A_{21}/A_{23} - A_{11}/A_{13})}{(A_{32}/A_{33} - A_{22}/A_{23})/(A_{31}/A_{33} - A_{21}/A_{23}) - (A_{22}/A_{23} - A_{12}/A_{13})/(A_{21}/A_{23} - A_{11}/A_{13})} \tag{2.16}$$

The still remaining x_3 is obtained e.g. from the equation 2.12 by dividing by A'_{i1}

$$x_3 = \frac{\bigwedge\limits_{Rows} \begin{bmatrix} (b_2/A_{22} - b_1/A_{12})/(A_{21}/A_{22} - A_{11}/A_{12}) \\ (b_3/A_{32} - b_2/A_{22})/(A_{31}/A_{32} - A_{21}/A_{22}) \end{bmatrix}}{\bigwedge\limits_{Rows} \begin{bmatrix} (A_{23}/A_{22} - A_{13}/A_{12})/(A_{21}/A_{22} - A_{11}/A_{12}) \\ (A_{33}/A_{32} - A_{23}/A_{22})/(A_{31}/A_{32} - A_{21}/A_{22}) \end{bmatrix}} = \frac{\bigwedge\limits_{i} b'_i/A'_{i1}}{\bigwedge\limits_{i} \left[A'_{i2}/A'_{i1}\right]} \tag{2.17}$$

and then forming the difference across the rows as usual.

$$x_3 = \frac{(b_3/A_{32} - b_2/A_{22})/(A_{31}/A_{32} - A_{21}/A_{22}) - (b_2/A_{22} - b_1/A_{12})/(A_{21}/A_{22} - A_{11}/A_{12})}{(A_{33}/A_{32} - A_{23}/A_{22})/(A_{31}/A_{32} - A_{21}/A_{22}) - (A_{23}/A_{22} - A_{13}/A_{12})/(A_{21}/A_{22} - A_{11}/A_{12})} \tag{2.18}$$

To check, one has to insert the determined x_j into the original system of equations 2.4. With the help of a program that can calculate symbolically, it is nowadays relativly quickly possible to verify the correctness of the result through matrix multiplication, which is why no further explanations are given here.
This example shows how one can get to any x_j you are looking for in almost any way. However, it seems that the effort is quite high since you have to start from the beginning more or less every time again. It is only recognizable that intermediate results that have already been calculated can be used to determine further x_j. Since in practice all x_j are generally of interest, the following section will show what the general procedure therefor should look like.

2.2 General matrix division

In section 2.1 the basic procedure or the basic structure of the algorithm of the matrix division is presented. However, the calculation of all x_j is associated with a high overall effort. In the following, a modified representation shall be chosen (this time simplified without the Δ-Sign) and based on this, the Gaussian algorithm shall be inferred. Starting with the general form

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix} / \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ A_{31} & A_{32} & \dots & A_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \tag{2.19}$$

follows in contrast to the previous example a programmatically more consistent procedure. One first divides by the first column, i.e. by the vector A_{i1},

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \frac{\begin{bmatrix} b_1/A_{11} \\ b_2/A_{21} \\ b_3/A_{31} \\ \vdots \\ b_n/A_{n1} \end{bmatrix}}{\begin{bmatrix} A_{11}/A_{11} & A_{12}/A_{11} & \dots & A_{1n}/A_{11} \\ A_{21}/A_{21} & A_{22}/A_{21} & \dots & A_{2n}/A_{21} \\ A_{31}/A_{31} & A_{32}/A_{31} & \dots & A_{3n}/A_{31} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1}/A_{n1} & A_{n2}/A_{n1} & \dots & A_{nn}/A_{n1} \end{bmatrix}} \tag{2.20}$$

and then forms the differences across the rows. In the first column below the fraction bar, it gets clear that this column becomes completely zero when “differentiating”. Just like it is also the goal of the Gauss method. In addition, this time the original first row from

equation 2.19 shall be retained, so that the following representation is obtained:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \frac{\begin{bmatrix} b_1 \\ b_2/A_{21} - b_1/A_{11} \\ b_3/A_{31} - b_2/A_{21} \\ \vdots \\ b_n/A_{n1} - b_{(n-1)}/A_{(n-1)1} \end{bmatrix}}{\begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ 0 & A_{22}/A_{21} - A_{12}/A_{11} & \dots & A_{2n}/A_{21} - A_{1n}/A_{11} \\ 0 & A_{32}/A_{31} - A_{22}/A_{21} & \dots & A_{3n}/A_{31} - A_{2n}/A_{21} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & A_{n2}/A_{n1} - A_{(n-1)2}/A_{(n-1)1} & \dots & A_{nn}/A_{n1} - A_{(n-1)n}/A_{(n-1)1} \end{bmatrix}} \tag{2.21}$$

Thus you get a sub-matrix $\boldsymbol{A}'$, which is recognizable from row 2 and column 2. In the same way, in the vector $\boldsymbol{b}$ from row 2 onwards a sub-vector $\boldsymbol{b}'$ has been generated.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \frac{\begin{bmatrix} b_1 \\ b'_2 \\ b'_3 \\ \vdots \\ b'_n \end{bmatrix}}{\begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ 0 & A'_{22} & A'_{23} & \dots & A'_{2n} \\ 0 & A'_{32} & A'_{33} & \dots & A'_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & A'_{n2} & A'_{n3} & \dots & A'_{nn} \end{bmatrix}} \tag{2.22}$$

The above steps are now repeated with the sub-matrix and the sub-vector, so first the sub-vector $\boldsymbol{b}'$ and all columns of the resulted sub-matrix $\boldsymbol{A}'$ are divided by the 1 column of this sub-matrix

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \frac{\begin{bmatrix} b_1 \\ b'_2/A'_{22} \\ b'_3/A'_{32} \\ \vdots \\ b'_n/A'_{n2} \end{bmatrix}}{\begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ 0 & A'_{22}/A'_{22} & A'_{23}/A'_{22} & \dots & A'_{2n}/A'_{22} \\ 0 & A'_{32}/A'_{32} & A'_{33}/A'_{32} & \dots & A'_{3n}/A'_{32} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & A'_{n2}/A'_{n2} & A'_{n3}/A'_{n2} & \dots & A'_{nn}/A'_{n2} \end{bmatrix}} \tag{2.23}$$

and then they get "differentiated" between the rows. The initial row 1 of the sub-matrix and sub-vector are left as they were.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \frac{\begin{bmatrix} b_1 \\ b'_2 \\ b'_3/A'_{32} - b'_2/A'_{22} \\ \vdots \\ b'_n/A'_{n2} - b'_{(n-1)}/A'_{(n-1)2} \end{bmatrix}}{\begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ 0 & A'_{22} & A'_{23} & \dots & A'_{2n} \\ 0 & 0 & A'_{33}/A'_{32} - A'_{23}/A'_{22} & \dots & A'_{3n}/A'_{32} - A'_{2n}/A'_{22} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & A'_{n3}/A'_{n2} - A'_{(n-1)3}/A'_{(n-1)2} & \dots & A'_{nn}/A'_{n2} - A'_{(n-1)n}/A'_{(n-1)2} \end{bmatrix}} \tag{2.24}$$

Once again a sub-vector $\boldsymbol{b}''$ and a sub-matrix $\boldsymbol{A}''$ are created.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \frac{\begin{bmatrix} b_1 \\ b'_2 \\ b''_3 \\ \vdots \\ b''_n \end{bmatrix}}{\begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ 0 & A'_{22} & A'_{23} & \dots & A'_{2n} \\ 0 & 0 & A''_{33} & \dots & A''_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & A''_{n3} & \dots & A''_{nn} \end{bmatrix}} \tag{2.25}$$

One can thus derive an algorithm that forms an upper triangular matrix by division and

difference.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{(n-1)} \\ x_n \end{bmatrix} = \frac{\begin{bmatrix} b_1 \\ b_2' \\ b_3'' \\ \vdots \\ b_{(n-1)}^{(n-2)} \\ b_n^{(n-1)} \end{bmatrix}}{\begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1(n-1)} & A_{1n} \\ 0 & A_{22}' & A_{23}' & \dots & A_{2(n-1)}' & A_{2n}' \\ 0 & 0 & A_{33}'' & \dots & A_{3(n-1)}'' & A_{3n}'' \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ddots & A_{(n-1)(n-1)}^{(n-2)} & A_{(n-1)n}^{(n-2)} \\ 0 & 0 & 0 & \dots & 0 & A_{nn}^{(n-1)} \end{bmatrix}} \tag{2.26}$$

Finally, it becomes apparent that the matrix division defined here in principle reflects the Gaussian algorithm. What's more, you can see that solving a system of linear equations is a kind of numerical derivation, since you differentiate between the equations until you arrive at the final result. The grade of the maximum derivative is equivalent to the rank of the matrix. This means that a matrix represents a multi-linear respectively non-linear function (more precisely a polynomial). A unique solution by matrix division exists if and only if this function is non-trivially differentiable or derivable $(n-1)$ times and the matrix thus has full rank, i.e. $rank(\boldsymbol{A}) = n$.

With the resulting upper triangular matrix, the entire vector x_j can finally be resolved by back substitution. So, in reverse order, one gets in succession:

$$\begin{bmatrix} x_n \\ x_{(n-1)} \\ \vdots \\ x_3 \\ x_2 \\ x_1 \end{bmatrix} = \begin{bmatrix} b_n^{(n-1)}/A_{nn}^{(n-1)} \\ (b_{(n-1)}^{(n-2)} - A_{(n-1)n}^{(n-2)} \cdot x_n)/A_{(n-1)(n-1)}^{(n-2)} \\ \vdots \\ (b_3'' - A_{3n}'' \cdot x_n - A_{3(n-1)}'' \cdot x_{(n-1)} - \dots)/A_{33}'' \\ (b_2' - A_{2n}' \cdot x_n - A_{2(n-1)}' \cdot x_{(n-1)} - \dots - A_{23}' \cdot x_3)/A_{22}' \\ (b_1 - A_{1n} \cdot x_n - A_{1(n-1)} \cdot x_{(n-1)} - \dots - A_{13} \cdot x_3 - A_{12} \cdot x_2)/A_{11} \end{bmatrix} \tag{2.27}$$

Back substitution causes the total solution to be split into a matrix division (matrix part below the diagonal) and a subsequent matrix multiplication (matrix part above the diagonal). Accordingly, this also splits the computing effort and reduces it drastically, since matrix multiplication is significantly less expensive than matrix division. As a reminder, if one wanted to find all x_j by matrix division, you would have to rotate the columns of the matrix $\boldsymbol{A}$ and repeat the matrix division $(n-1)$ times so that a different x_j is found with each recalculation. This was sufficiently discussed in chapter 2.1. For clarification see also the Matlab-Code in the appendix 4.1.
It remains to be mentioned that in the practical application of matrix division, of course division by zero can occur, which is why appropriate precautionary measures must be taken when programming. When programming, it is also easy to see that with small

changes the matrix division algorithm can be converted into the Gaussian algorithm and vice versa. Compare for this the pseudo-code in the Wikipedia article on the Gaussian elimination method [1] and the Matlab-Code in the appendix 4.2.

2.3 Treatment of rectangular matrices

To complete the matrix division, it must be briefly discussed what happens if, contrary to the assumption in the chapter 2.2, the matrix is not square but rectangular or does not have full rank.

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_m \end{bmatrix} / \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ A_{31} & A_{32} & \dots & A_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1} & A_{m2} & \dots & A_{mn} \end{bmatrix} \tag{2.28}$$

If $m > n$ this means that one can form differences more often than divide because there are more equations than unknowns. So there are more rows than columns, a so-called overdetermined system. For more information see the website of Dankert/Dankert [2] or the Youtube videos by Gilbert Strang [3].
In practice, such systems of equations arise, for example, by assuming a function with n unknowns for a relationship that is actually more complex. As compensation, more test values m are collected and the system of equations is solved by the method of least squares. When inserting the approximate solution into the original equation, one can then determine the individual deviations or errors from the original vector $\boldsymbol{b}$. Based on this, one can decide whether the underlying function is sufficiently accurate for your own purposes and adjust it if necessary.
Alternatively, one could solve the system of equations with matrix division only as far as the square matrix contained therein with $rank(\boldsymbol{A}) = n$ allows. If the remaining rows i with $m \geq i > n$ in the vector $\boldsymbol{b}$ have not become zero, it is clear that the underlied function has too few unknowns. One could define an initial solution in which an error is accepted, which one would then have to specify in the solution. This solution solves the first n rows exactly. The remaining rows then reveal the unresolvable error, similar to a scalar division where an unresolved remainder is given instead of decimal places.

If $n > m$ this means one can't take differences as often as one can divide because there are fewer equations than unknowns. So there are fewer rows than columns, a so-called underdetermined system.
In this case there are infinitely many solutions, of which one could utilize the most trivial one as the first solution. One would therefore solve the system of equations as long as the square matrix contained therein with $rank(\boldsymbol{A}) = m$ allows. The excessing columns j with $n \geq j > m$ would have to be excluded respectively the corresponding x_j would have to be set to zero. This gives a well-defined exact solution that thoroughly can be put to meaningful use.

3 Conclusion

The matrix division defined in this work certainly does not reveal all too major new insights, but the agreement could go hand in hand that the matrix division verifiably exists and we have always applied it implicitly using the Gaussian algorithm. Even if the form and rank of the matrix determine how accurate the result looks and to what extent it can be further used.

4 Appendix

The following code examples are the implementation of matrix division. They are intended for verification and comparison purposes.

4.1 Matlab-Code: Matrix division (original)

```
function [x] = mdiv(A,b)
% Attention:  Suitable for symbolic calculation,
% but division by zero possible!

% Input:
    % A   % Matrix A
    % b   % Vector b

% Output:
    % x   % Solution Vector x

% Algorithm:
    x = b;
    n = length(b);
    for i = 1 : n
            R = [A, b];
            S = R(:, n); R(:, n) = R(:, i); R(:, i) = S;  % Column Swapping
            for j = 1 : n
                for k = 2 : n - j + 2
                    R(:, k) = R(:, k)./R(:, 1);
                end

                % System Reduction
                R = R(:, 2 : k);
                if length(R) > 1
                    R = diff(R);
                else
                    x(i) = R;
                end
            end
    end

return
```

4.2 Matlab-Code: Matrix division (optimized)

```
function [x] = mdiv(A,b)
% Attention:  Suitable for symbolic calculation,
% but division by zero possible!

% Input:
    % A   % Matrix A
    % b   % Vector b

% Output:
    % x   % Solution Vector x

% Algorithm:
    R = [A, b];
    n = length(b);
    for j = 1 : n - 1
        for i = n : -1 : j + 1
            R(i,:) = R(i,:)/R(i,j) - R(i-1,:)/R(i-1,j);
        end
    end

    % Back Substitution
    b = R(:, n + 1);
    x = b;
    for i = n : -1 : 1
        x(i) = (b(i) - R(i, n : -1 : i + 1) * x(n : -1 : i + 1))./R(i,i);
    end

return
```

References

[1] Wikipedia: *Gaussian elimination* | www.wikipedia.org

[2] Dankert/Dankert: *Mathematik für die Technische Mechanik* | www.tm-mathe.de

[3] Gilbert Strang: *MIT 18.06 Linear Algebra, Spring 2005* | www.youtube.com